Claim 61.   (Unmodified)

A method of communication between a first object located on a first computer and a second object located on a second computer, the first and second computers connected by a network, the method comprising:

calling an interface of the second object by the first object on the first computer, wherein the interface of the second object is identified only with an interface pointer identifier, and wherein the calling the interface of the second object by the first object comprises bypassing a mechanism, the bypassed mechanism comprising adding a remote procedure call interface identifier to the call;

performing remote procedure call utility functions on the call at the first computer; and

communicating the call to the second computer, wherein the second computer receives the call, performs remote procedure call utility functions on the call, passes the call to a dispatching function so as **to bypass a remote procedure call dispatching function**, invokes a stub, and accesses the interface of the second object identified by the interface pointer identifier.

Claim 119. (Differences shown from claim 61)

A method of communication between a first object located on a first computer and a second object located on a second computer, the first and second computers connected by a network, the method comprising:

calling an interface of the second object by the first object on the first computer~~, wherein the interface of the second object is identified only with an interface pointer identifier~~, and wherein the calling the interface of the second object by the first object comprises <u>(a)</u> bypassing a mechanism, the bypassed mechanism comprising adding a remote procedure call <u>(RPC)</u> interface identifier <u>(IID) of the second object</u> to the call<u>, and (b) adding an interface pointer identifier (IPID) to the call</u>;

performing ~~remote procedure call~~ <u>RPC</u> utility functions on the call at the first computer; and

communicating the call to the second computer, wherein the second computer receives the call, performs ~~remote procedure call~~ <u>RPC</u> utility functions on the call, passes the call to a dispatching function so as to bypass a remote procedure call dispatching function, invokes a stub, and accesses the interface of the second object identified by the ~~interface pointer identifier~~ <u>IPID</u>.

Claim 120. (Differences shown from claim 119)

A method of communication between a first object located on a first computer and a second object located on a second computer, the first and second computers connected by a network, the method comprising:

calling an interface of the second object by the first object on the first computer, and wherein the calling the interface of the second object by the first object comprises (a) bypassing a mechanism, the bypassed mechanism comprising adding a remote procedure call (RPC) interface identifier (IID) of the second object to the call, and (b) adding an interface pointer identifier (IPID) to the call;

performing RPC utility functions on the call at the first computer; and

communicating the call to the second computer, wherein the second computer:

receives the call~~,~~:

performs ~~remote procedure call~~ RPC utility functions on the call~~,~~:

determines if the call includes the IPID:

if the call does not include the IPID, calls an RPC dispatching function:

if the call does include the IPID, calls an alternative dispatching function based on the IPID, bypassing the RPC dispatching function, ~~passes the call to a dispatching function so as to bypass a remote procedure call dispatching function,~~ invokes a stub, and accesses the interface of the second object identified by the ~~interface pointer identifier~~ IPID.

Claim 121. (Differences shown from claim 61)

A method of communication, using a remote procedure call (RPC) model, between a first object located on a first computer and a second object located on a second computer, the first and second computers connected by a network, the method comprising:

calling an interface of the second object by the first object on the first computer, ~~wherein the interface of the second object is identified only with an interface pointer identifier, and~~ wherein the calling the interface of the second object by the first object comprises bypassing a mechanism, the bypassed mechanism comprising adding a RPC ~~remote procedure call~~ interface identifier (IID) of the second object to the call;

performing RPC ~~remote procedure call~~ utility functions on the call at the first computer; and

communicating the call to the second computer, wherein the second computer receives the call, performs RPC ~~remote procedure call~~ utility functions on the call, passes the call to a dispatching function so as to bypass a RPC ~~remote procedure call~~ dispatching function, invokes a stub, and accesses the interface of the second object identified by the interface pointer identifier.

Claim 122.   (Differences shown from Claim 121)

A method of communication, using a remote procedure call (RPC) model, between a first object located on a first computer and a second object located on a second computer, the first and second computers connected by a network, the method comprising:

calling an interface of the second object by the first object on the first computer, wherein the calling the interface of the second object by the first object comprises bypassing a mechanism, the bypassed mechanism comprising adding a RPC interface identifier (IID) of the second object to the call, wherein the calling the interface of the second object by the first object further comprises providing an interface pointer identifier (IPID) of the second object to the call;

performing RPC utility functions on the call at the first computer; and

communicating the call to the second computer, wherein the second computer receives the call, performs RPC utility functions on the call, passes the call to a dispatching function so as to bypass a RPC dispatching function, invokes a stub, and accesses the interface of the second object identified by the interface pointer identifier.